#14
LDJ
11-19-02

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Lita** § Group Art Unit: **2155**
§
Serial No.: **09/282,692** § Examiner: **Duong, O.**
§
Filed: **03/31/1999** § Atty. Docket No.: **AT9-98-700**
§

For: **Method and System for Using Virtual URLs for Load Balancing (as amended)**

Certificate of Mailing Under 37 C.F.R. § 1.8(a)

I hereby certify that this correspondence is being deposited with the United States Postal Service as First Class mail in an envelope addressed to:
BOX AF
Assistant Commissioner of Patents
Washington, D.C. 20231
on November 12, 2002.

By: Joseph R. Burwell
Joseph R. Burwell, Reg. No 44,468

BOX AF
Assistant Commissioner of Patents
Washington, D.C. 20231

APPELLANT'S BRIEF
IN RESPONSE TO OFFICE ACTION UNDER 37 C.F.R. § 1.192

This brief is filed in triplicate in support of the previously filed Notice of Appeal, which has an Office date of receipt of 09/09/2002, and which appealed from the decision of the examiner dated 05/08/2002 rejecting claims 1-22. The fee required under 37 C.F.R. § 1.17(c) for filing a brief in support of an appeal is provided in the Transmittal of Appeal Brief filed herewith.

11/18/2002 AWONDAF1 00000069 090447 09282692
01 FC:1402 320.00 CH

1. REAL PARTY IN INTEREST

The real party in interest in this appeal is International Business Machines Corporation (IBM).

2. RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no such appeals or interferences.

3. STATUS OF CLAIMS

Claims 1-22 are pending in this application; claims 1-22 have been finally rejected; claims 1-22 have been appealed. No claims have been canceled, withdrawn, or allowed.

4. STATUS OF AMENDMENTS

No after-final amendments have been filed.

5. SUMMARY OF INVENTION

The present invention provides a method for distributing client requests across a pool of servers on a per-session basis rather than on a per-connection basis. A server in the pool of servers is allocated a number of sessions such that a client's HTTP connection requests are handled by the same server throughout a user session. A load balancing routine across a set of servers ensures that connection requests from a client during its session are serviced by the same server.

In response to a connection request, a front-end managing server intercepts the request and recognizes that the request will initiate a user session. The managing server acts as a redirector for connection requests; the managing server may query a load balancing routine to determine which server in the pool of servers should service the new session. A unique session identifier is associated with a given server in the pool of servers, and the session identifier is then incorporated into a base URL for the assigned server, thereby forming a "virtual URL" that is returned in an appropriate redirection response to the client, which then automatically issues a new HTTP connection request using the newly generated virtual URL. All subsequent data to the client will incorporate the virtual URL such that subsequent requests from the client will contain the session identifier as part of the URL. Client requests are then routed to the appropriate server in accordance with the URL, and the server can use the session identifier to associate requests with a user session.

6. ISSUES

The issues on appeal are:

whether claims 1-4, 6-10, 12-16, 18, 19, 21, and 22 are unpatentable over Bayeh et al., "Maintaining Sessions in a Clustered Server Environment", U.S. Patent Number 6,098,093, filed 03/19/1998, issued 08/01/2000, in view of Narendran et al., "Data Distribution Techniques for Load-Balanced Fault-Tolerant Web Access", U.S. Patent 6,070,191, filed 10/17/1997, issued 05/30/2000; and whether claims 5, 11, 17, and 20 are unpatentable over Bayeh et al. in view of Narendran et al. and further in view of Brodd et al., "Network Communications Interface", U.S. Patent 5,303,238, filed 12/11/1990, issued 04/12/1994.

7. GROUPING OF CLAIMS

The claims stand and fall together as a single group.

8. ARGUMENTS

Was 35 U.S.C. § 103(a) properly applied in a rejection of claims 1-22 as being unpatentable over Bayeh et al. in view of Narendran et al., and further in view of Brodd et al.?

Standards for *prima facie* case of obviousness

Appellant cites the following from MPEP Chapter 2100:

> The examiner bears the burden of establishing a *prima facie* case of obviousness. To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, and not based on applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991).
>
> The initial burden is on the examiner to provide some suggestion of the desirability of doing what the inventor has done. "To support the conclusion that the claimed invention is directed to obvious subject matter, either the references must expressly or impliedly suggest the claimed invention or the examiner must present a convincing line of reasoning as to why the artisan would have found the claimed invention to have been obvious in light of the teachings of the references." *Ex parte Clapp*, 227 USPQ 972, 973 (Bd. Pat. App. & Inter. 1985).
>
> When the motivation to combine the teachings of the references is not immediately apparent, it is the duty of the examiner to explain why the combination of the teachings is proper. *Ex parte Skinner*, 2 USPQ2d 1788 (Bd. Pat. App. & Inter. 1986). A statement of a rejection that includes a large number of rejections must explain with reasonable specificity at least one rejection, otherwise the examiner procedurally fails to establish a *prima facie* case of obviousness. *Ex parte Blanc*, 13 USPQ2d 1383 (Bd. Pat. App. & Inter. 1989).

Arguments in support of patentability

With respect to the second ground of rejection of unpatentability over Bayeh et al. in view of Narendran et al. and further in view of Brodd et al., the rejection states that the combination of Bayeh et al. and Narendran et al. does not teach the inactivation of a session identifier, and the rejection of these claims then relies on Brodd et al. as disclosing the inactivation of a session identifier. However, Bayeh et al. states at column 12, lines 63, that a session may become invalid. Hence, Appellant asserts that Bayeh et al. discloses at least as much as Brodd et al. and that the rejection need not rely upon a third reference. Hence, for this reason and the reasons provided below for the grouping of claims, Appellant does not include any specific arguments directed to this ground of rejection.

With respect to the grouping of claims for arguments in support of patentability, independent claims 1, 9, 21, and 22 are method claims, whereas independent claim 15 is directed to a computer program product, and independent claim 18 is directed to a server. Claims 15 and 18 correspond in form with claim 9 and may be considered as similar to claim 9 for the purposes of this argument.

Although method claims 1, 9, 21, and 22 are similar, claim 1 is broader than claims 9, 21, and 22. Claim 9 differs from claim 1 in that claim 9 mentions the use of a load balancing protocol (as does claim 21), whereas claim 22 mentions the use of a particular form of session identifier. In other words, independent claim 1 is the broadest independent claim. Hence, for purposes of this argument,

Appellant argues for the patentability of the present invention using claim 1 as an exemplary claim.

Rejections under 35 U.S.C. 103 must provide a *prima facie* case for obviousness as described in the above-cited section of MPEP 2100. According to 37 C.F.R. § 1.192(c)(8)(iv), for each rejection under 35 U.S.C. § 103, Appellant must specify the errors in the rejection, the specific limitations in the rejected claims which are not described in the prior art relied on in the rejection, and how such limitations render the claimed subject matter nonobvious over the prior art. If the rejection is based upon a combination of references, the argument shall explain why the references, taken as a whole, do not suggest the claimed subject matter. In summary, Appellant argues that the pending claims in the present patent application are patentable because the rejection of the independent claim 1 fails to provide a *prima facie* case of obviousness.

Initial review of the teachings of *Bayeh et al.*

In its background section, Bayeh et al. explains that session identifiers have been implemented within HTTP communications as a method for managing state information, even though HTTP is defined as a stateless protocol. There have been two primary approaches: cookies and URL rewriting. In the first case, a server can generate a cookie in response to a client request, and the cookie contains a session identifier. The session cookie is passed back to the client, and the cookie is returned by the client with each subsequent request to the server. When the server receives a request

with a cookie, the session identifier within the cookie enables the server to find information about previous transactions for the client, thereby allowing the server to maintain state information about the client. In the second approach, URL rewriting is used to ensure that requests sent to the server will have the session identifier in the URL of a request. When a Web page is generated in response to a client request, the hypertext links that are embedded in the Web page are modified to contain the session identifier for the requesting client. When a user at the client clicks on one of these hypertext links, the URL in the request that is returned to the server will contain the client's session identifier. The system described in Bayeh et al. can use either session cookies or URL rewriting.

Bayeh et al. mentions that a common practice for scaling and increasing the capacity of Web servers and Web sites is to increase the number of computer hosts (servers) which perform HTTP processing. The system that is disclosed in Bayeh et al. falls into this category of solutions; Bayeh et al. teaches a system in which session-related state information is managed in a clustered server environment. Bayeh et al. also states a particular purpose for its disclosed system in its background section at column 4, line 65, to column 5, line 10:

> For example, if a client request is received at one server, and that server maintains information about the on-going session, there is no way for this version of the session information to be made available to a different server in the cluster if the next request from this client goes to a different server. Accordingly, a need exists for a technique by which these shortcomings in the ability to provide session services in a clustered environment can be overcome.

Bayeh et al. describes the system as follows at column 8, lines 42-58 (emphasis added):

> FIG. 3 illustrates a model of the clustered server environment in which the present invention may be practiced, and shows how this invention interacts with other components in the environment. A Web server 60 may be connected to any number of other Web servers, shown here as 62 and 64. Clustering multiple servers in this way provides for increased capacity with which HTTP requests at a Web site can be processed. A load-balancing host 59 functions as a type of front-end processor to these servers, receiving client requests 100, 101, 102 and then routing these requests (shown here as 110, 111, 112) to a server selected according to policies implemented in the load-balancing host software. Note that the requests 110, 111, 112 are shown being sent to specific Web servers; **this is merely an example of a possible outcome of the load balancing process**. Load-balancing techniques are known in the art and do not form part of the present invention.

The system taught by Bayeh et al. allows the load-balancing host to function without modification. The load-balancing host acts as a front-end processor for the cluster of servers to route client requests to the servers in accordance with a load-balancing routine at the host. As should be understood by the emphasized portion, there is no mechanism to ensure that client requests from a particular client for a particular user session are routed by the load-balancing host to a particular server or servlet. In other words, the load-balancing host does not provide session management services.

The system taught by Bayeh et al. solves the problem of extending session management across a cluster of servers by providing session services via a set of plug-in servlet

engines. One of these servlet engines will be installed on each Web server in the cluster of servers, and one of the servlet engines is configured to function as a session management server while the other servlet engines are configured to function as session clients. When an HTTP request is received from a client, the request is sent by the load-balancing host to one of the Web servers. The Web server then passes the request to the plug-in servlet engine based on certain criteria, such as the presence of a session identifier string as part of the host destination address in the URL. In other words, different servlets perform different, possibly application-specific, functions, and the necessary servlet may be indicated within the URL within the client request.

Bayeh et al. then continues to describe the system as follows in column 10, lines 10-31:

> When the plug-in servlet engine 72 gets the request 112, the request may or may not include a session identifier for a session with this client. If this is the first request of a new session, no session ID will be present. If this is a request of an existing session, then there will be a session ID included using either the cookie mechanism or URL rewriting, as discussed earlier.

Bayeh et al. also describes the manner in which session information is shared among servers and servlets at column 11, lines 3-8:

> When the servlet processing is finished, the session object is returned to the session pool, where it can be accessed for subsequent transactions with this servlet or a different servlet in the clustered environment. In this way, the state of the session can be communicated among the clustered servers and their servlets.

As can be understood from the description above, a client request is received by a load-balancing host, which then forwards the client request to a Web server in a cluster of Web servers in accordance with a load-balancing algorithm. Assuming that the client request does not contain a session identifier using either a cookie or URL rewriting technique, then the plug-in servlet client at the receiving Web server coordinates with the plug-in servlet server to generate a session object for the user session associated with the client request. Subsequent requests from the client during the same user session will contain the session identifier, and the session identifier can be used to manage information for the user session across many client requests or connections.

Initial review of the teachings of *Narendran et al.*

The system disclosed in Narendran et al. is fairly summarized by its abstract:

> A server system for processing client requests received over a communication network includes a cluster of N document servers and at least one redirection server. The redirection server receives a client request from the network and redirects it to one of the document servers, based on a set of pre-computed redirection probabilities. Each of the document servers may be an HTTP server that manages a set of documents locally and can service client requests only for the locally-available documents. A set of documents are distributed across the document servers in accordance with a load distribution algorithm which may utilize the access rates of the documents as a metric for distributing the documents across the servers and determining the redirection probabilities. The load distribution algorithm attempts to equalize the sum of the access rates of all the documents stored at a given document server across all of the document servers. In the event of a server failure, the redirection

probabilities may be recomputed such that the load of client requests is approximately balanced among the remaining document servers. The redirection probabilities may also be recomputed periodically in order to take into account changes in document access rates and changes in server capacity. The recomputation may be based on a maximum-flow minimum-cost solution of a network flow problem.

The redirection process that is used by the system disclosed in Narendran et al. is explained in the portion of the reference that is cited by the rejection and a subsequent portion:

> As described above, the present invention utilizes a redirection mechanism for achieving load balance among a cluster of document servers. The redirection mechanism in the illustrative embodiments is an integral part of the HTTP protocol and is supported by all browsers and web servers. Alternative embodiments of the invention may utilize a redirection mechanism implemented at a higher level, such as redirection at the router level based on Internet Protocol (IP) addresses. In an HTTP redirection embodiment, if a client request received at a redirection server is to be redirected to another server, the original redirection server sends a redirection message to the client. The redirection message typically contains the URL or other identifier of the new server. The client then makes a separate request to the new server. The mapping which dictates that a URL should be redirected to another server may be located in the configuration file, which can be identified by a .conf suffix. Since a document may be replicated on more than one server, an incoming request for the document can be mapped to any of the servers on which the document exists. In the system 10 of FIG. 1, a request for a document is directed by the redirection server 14-1 or 14-2 to one of the document servers in server cluster 16 with a predetermined probability. This probability is determined by the document distribution algorithm described above. A configuration file htd.conf may specify the mapping of a URL to multiple URLs. For a given URL, each document server which is capable of

> serving the document associated with the URL, has a probability associated with it. This probability can be specified along with the document mapping in the configuration file. ... Using these probabilities, the redirection server chooses one document server and sends a redirect message with the relevant URL to the client, which then connects directly to the document server using this URL.
> --[Narendran et al., column 14, lines 40, to column 15, line 2; column 15, lines 22-24]

The system disclosed by Narendran et al. distributes copies of documents across multiple servers, and clients may request those documents. When a client's request is received by a redirection server, the request may be redirected by the redirection server back through the client to a server that should provide the requested document. In one embodiment, Narendran et al. accomplishes the redirection operation using the well-known redirection mechanism in the HTTP protocol.

Contrasting the present invention and *Bayeh et al.*

Three important distinctions can be made between the system of the present invention and the system described by Bayeh et al.. In the present invention, the front-end managing server participates in the session management, and client requests are redirected from the front-end managing server back through the client to a server in the pool of servers. In addition, the present invention ensures that the same server in the pool of servers receives all of the client requests for a particular user session.

In contrast, the load-balancing host in the system taught by Bayeh et al. does not participate in the session management services among the cluster of Web servers, and the client

request is forwarded directly from the load-balancing host to any of the Web servers. In addition, once a session identifier has been assigned to a user session in the system taught by Bayeh et al., a client request within a user session may be received at any of the servers in the cluster of servers; this is facilitated by the fact that the plug-in servlet clients at each of the servers coordinate the session management information amongst themselves with the assistance of the plug-in servlet server. In other words, there is no guarantee that the same server will process all client requests for a given user session. Hence, the load on the cluster of servers is balanced on a per-connection basis.

Contrasting the present invention and *Narendran et al.*

In a manner similar to that observed above with respect to Bayeh et al., two important distinctions can be made between the system of the present invention and the system described by Narendran et al.. In the present invention, the front-end managing server coordinates session management, and the present invention ensures that the same server in the pool of servers receives all of the client requests from a particular client for a particular user session.

In contrast, the load-balancing, redirection server in the system taught by Narendran et al. does not coordinate session management services among the cluster of Web servers. Before or after redirecting a client's request back through the client, there is no attempt by the redirection server to coordinate a client session that persists across multiple requests from the same client. Hence, once a client request

has been redirected back through the client in the system taught by Narendran et al., the next request from the same client may be received at any of the servers in the cluster of servers because the redirection server chooses a server based on whether or not the server has a copy of the requested document. The redirection server does not attempt to send all of the requests from a particular client to the same server. In other words, there is no guarantee that the same server will process all client requests for a given user session. Hence, the load on the cluster of servers is balanced on a per-connection basis.

In fact, if the redirection server did attempt to send all of the requests from a client to the same server, there would be no pre-determination that the server had a copy of the requested document as is required by the successful operation of the system. This fact is reinforced by a statement within the portion of Narendran et al. that was recited above, specifically, at column 14, lines 57-59:

> Since a document may be replicated on more than one server, an incoming request for the document can be mapped to any of the servers on which the document exists.

In the hypothetical scenario in which the redirection server sends all of the requests from a client to the same server, the actions of the redirection server would guarantee that some of the requests would fail, which should be intolerable to anyone who would implement this hypothetical system.

Contrasting independent claim 1 with the prior art

Independent claim 1 reads:

> 1. A method for managing connection requests to a pool of servers identified by a given URL, comprising the steps of:
>
> in response to a connection request from a given client machine that initiates a session, associating a session identifier with a given server in the pool;
>
> using the session identifier in a redirection response;
>
> returning the redirection response to the given client to redirect the connection request to the given server; and
>
> during the session, receiving at the given server any additional connection requests from the given client machine.

In the rejection of independent claim 1, Bayeh et al. is used as a primary reference, and Narendran et al. is used as a secondary reference. As discussed above, Bayeh et al. discloses a system that uses session identifiers within a session management scheme, while Narendran et al. discloses a well-known method for redirecting client requests using a redirection mechanism within the HTTP protocol.

With respect to the rejection of independent claim 1, the rejection applies Bayeh et al. to the first, second, and fourth elements of claim 1. With respect to the first element, i.e. "in response to a connection request from a given client machine that initiates a session, associating a session identifier with a given server in the pool", Bayeh et al. discloses the creation and use of session identifiers, but Bayeh et al. does not associate a session identifier with a given server in the pool of servers. As noted above, Bayeh et al. specifically states that the disclosed system attempts to

provide a solution to the fact that a next request from a particular client during a particular session may be routed by a load-distributing host across multiple different servers. A server that receives a client request supports servlets, and these servlets interact with a special servlet that acts as a session server that maintains the session objects. Hence, Bayeh et al. does not disclose "associating a session identifier with a given server".

Moreover, the rejection is misleading on this point. The rejection states that the first element of claim 1 is disclosed by a portion of Bayeh et al. at column 3, lines 40-42, which reads as follows: "URL rewriting is a way of ensuring that requests sent to the server will have the session ID plugged into the URL." As one can see, the cited passage does not disclose the feature for which it was cited by the rejection. As explained above, the load-balancing host in the system of Bayeh et al. executes a load-balancing algorithm without using the session identifier; the load-balancing host does not use the session ID in any manner. A client request may be directed to any server, after which a servlet at the server may use the session ID to retrieve a session object that contains session information for a client session. Again, Bayeh et al. does not disclose "associating a session identifier with a given server", as required by the claim language.

With respect to the second element of claim 1, i.e. "using the session identifier in a redirection response", Bayeh et al. simply does not disclose a redirection response. Moreover, the rejection is also misleading on this point. The

rejection states that the second element of claim 1 is disclosed by a portion of Bayeh et al. at column 3, lines 46-53, which reads as follows:

> By putting the session ID into the address in that link, the server can maintain state information for the session. Processing by the server is required for rewriting the URLs. That is, before the server sends a page to a client, the server will check to see if the page has URLs embedded in it. If so, the server adds a session ID parameter and the unique identifier for this session into the URL syntax before sending the page.

As should be apparent, there is no mention of a redirection response in the cited passage.

With respect to the fourth element of claim 1, i.e. "during the session, receiving at the given server any additional connection requests from the given client machine", Bayeh et al. does not disclose this feature. As explained above, in the system disclosed by Bayeh et al., the load on the cluster of servers is balanced on a per-connection basis, and a next request from a particular client during a particular session may be routed by a load-distributing host to one of any number of different servers.

Again, the rejection is also misleading on this point. The rejection states that the fourth element of claim 1 is disclosed by a portion of Bayeh et al. at column 4, line 51, to column 5, line 3, the relevant portion of which was already recited above. In particular, this passage states: "If a client request is received at one server, and that server maintains information about the on-going session, there is no way for this version of the session information to be made available to a different server in the cluster if the next

request from this client goes to a different server." Hence, the cited passage does not disclose the feature for which the rejection relies upon the passage as disclosing. More importantly, the passage actually refutes the point that the rejection attempts to make.

With respect to the third element of claim 1, i.e. "returning the redirection response to the given client to redirect the connection request to the given server", the rejection admits that "Bayeh does not disclose returning the redirection response to the given client to redirect the connection request to the given server."--(Office action, page 2, last paragraph). In order to remedy this deficiency, the rejection relies on a portion of Narendran et al. that was recited above. However, as discussed above, Narendran et al. does not disclose the use of session identifiers, as required by the second element of claim 1, so Narendran et al. cannot be relied upon as disclosing an equivalent manner of using redirection responses as used within the present invention.

More importantly, as explained above, Narendran et al. does not disclose session management in any manner whatsoever, and Bayeh et al. does not disclose redirection responses in any manner whatsoever. The rejection merely asserts that it would have been obvious to join the features of the system disclosed in Bayeh et al. and the system disclosed in Narendran et al. in a hypothetical combination without any explanation. The motivational statement in the rejection merely states the following in the first paragraph on page 3:

> Therefore, it would have been obvious to have used the returning the redirection response in Bayeh as taught by Narendran because it would enable the client to directly connect to the assigned server in the pool of servers using the specified URL without passing through the redirector so as to increase the efficiency of the system.

Appellant strongly disagrees with this statement. The rejection does not mention which entity would be responsible for redirecting the client request. If it is the load-balancing host in the system of Bayeh et al., and the load-balancing host does not perform session management, then the load-balancing host could not ensure that all requests from a given client are received at the same server during a particular session, as is explicitly required by the claim language of claim 1. Yet no other entity could reasonably be conjectured as performing the redirection operation in the hypothetical combined system. If one were to assert that one of the servers in Bayeh et al. performed the redirection operation, then one would have to ask why a server that could satisfy the request would be redirecting the request to another server, which would be nonsensical.

Appellant asserts that the motivational statement is extremely misleading and incongruous. The load-balancing host in the system of Bayeh et al. is utilized to enhance the efficiency of the system; all client requests are received at the load-balancing host, which then disperses the requests among a cluster of servers in accordance with a load-balancing algorithm. The motivational statement states that employing the redirection technique of Narendran et al. would enhance the efficiency of a hypothetical combined system because a

client could communicate directly with a server. If this were the case, then the load-balancing host of Bayeh et al. would not be able to perform its duties, and there would be no need for the load-balancing host, yet the structure of the server environment in Bayeh et al. requires the load-balancing host.

No matter what perspective is used by one of ordinary skill in the art, the suggested combination in the rejection would destroy a substantial advantage in the operation of the system in Bayeh et al.. Hence, the rejection is incorrect in its conclusion that one of ordinary skill in the art would have been motivated to combined features from the primary and secondary references. This argument is supported by the MPEP, which states the following within MPEP § 2143.02:

> If the proposed modification would render the prior art invention being modified unsatisfactory for its intended purpose, then there is no suggestion or motivation to make the proposed modification. *In re Gordon*, 733 F.2d 900, 221 USPQ 1125 (Fed. Cir. 1984).

In addition, the suggested combination in the rejection would require a substantial change in the operation of the system in Bayeh et al.. Again, the rejection is incorrect in its conclusion that one of ordinary skill in the art would have been motivated to combine features from the primary and secondary references. This argument is supported by the MPEP, which states the following within MPEP § 2143.01:

> If the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render the claims *prima facie* obvious. *In re Ratti*, 270 F.2d 810, 123 USPQ 349 (CCPA 1959).

Hence, it would not be possible to modify Bayeh et al. to include a redirection response without rendering Bayeh et al. inoperable for its intended use.

Examiner bears the burden of establishing a *prima facie* case of obviousness

Bayeh et al. clearly fails to show features of the present invention as currently claimed and as improperly asserted by the Office action, thereby rendering Bayeh et al. incapable of being used as a primary reference as argued by the current rejection. Moreover, both Narendran et al. and the combination of Bayeh et al. and Narendran et al. fail to show the claimed features. As should be recognized, because both the primary and secondary references in the rejection fail to disclose the claimed features against which the references were applied, and because the references fail to be combinable to produce this feature, the rejection fails to fulfill the requirements of a proper obviousness argument.

Hence, a rejection of the claims cannot be based upon the cited prior art to establish a *prima facie* case of obviousness. Therefore, a rejection of the claims under 35 U.S.C. § 103(a) has been shown to be insupportable in view of the cited prior art, and the claims are patentable over the applied references. For this and other reasons, Appellant argues that the position of the Examiner should be reversed and that the rejection of the claims should be withdrawn.

9. Conclusion

In view of the above arguments, it is respectfully urged that the rejection of the claims should not be sustained.

DATE: November 12, 2002

Respectfully submitted,

Joseph R. Burwell
Reg. No. 44,468
ATTORNEY FOR APPELLANT

Law Office of Joseph R. Burwell
P.O. Box 28022
Austin, Texas 78755-8022
Voice: 866-728-3688 (866-PATENT8)
Fax: 866-728-3680 (866-PATENT0)
Email: joe@burwell.biz

10. APPENDIX OF CLAIMS

1. A method for managing connection requests to a pool of servers identified by a given URL, comprising the steps of:

in response to a connection request from a given client machine that initiates a session, associating a session identifier with a given server in the pool;

using the session identifier in a redirection response;

returning the redirection response to the given client to redirect the connection request to the given server; and

during the session, receiving at the given server any additional connection requests from the given client machine.

2. The method as described in claim 1 wherein the step of using the session identifier includes generating a virtual URL.

3. The method as described in claim 2 wherein the virtual URL comprises a URL in the connection request modified to include the session identifier.

4. The method as described in claim 1 wherein the session identifier is incorporated in data returned from the given server to the client machine.

5. The method as described in claim 1 further including the step of:

in response to a connection request from the given client machine that terminates the session, inactivating the session identifier.

6. The method as described in claim 1 wherein the given client machine include a browser.

7. The method as described in claim 1 wherein each of the servers in the pool supports a similar set of objects.

8. The method as described in claim 1 wherein the session identifier is associated with a given server as a function of a load balancing protocol.

9. A method for managing connection requests to a pool of servers, comprising the steps of:

responsive to a connection request from a client machine to initiate a user session, associating a user session originating from a client machine with a given server in the pool in accordance with a load balancing protocol;

returning a redirection response to the client machine for the connection request; and

during the user session, receiving at the given server any additional connection requests originating from the client machine.

10. The method as described in claim 9 wherein the associating step comprises:

generating a virtual URL by modifying a given URL to include a session identifier;

using the virtual URL to redirect the connection request to the given server.

11. The method as described in claim 10 further including the step of:

inactivating the virtual URL upon completion of the user session.

12. The method as described in claim 10 wherein all data returned from given server to the client machine includes the session identifier.

13. The method as described in claim 9 wherein each of the servers in the pool supports a similar set of given objects.

14. The method as described in claim 9 wherein each client machine include a Web browser.

15. A computer program product in a computer-readable medium for managing connection requests to a pool of servers, comprising the steps of:

means, responsive to a connection request from a client machine to initiate a user session, for associating a user session originating from a client machine with a given server in the pool in accordance with a load balancing protocol;

means for returning a redirection response to the client machine for the connection request; and

means operative during the user session for receiving at the given server any additional connection requests originating from the client machine.

16. The computer program product as described in claim 15 wherein the associating means comprises:

means for generating a virtual URL by modifying a given URL to include a session identifier;

means for redirecting a given connection request to the given server using the virtual URL.

17. The computer program product as described in claim 16 further including:

means for inactivating the virtual URL upon completion of the user session.

18. A server for managing a pool of servers at a Web site identified by a given URL, comprising:

a processor;

an operating system;

a load balancing routine; and

a redirector routine for managing HTTP connection requests to the Web site, comprising:

means responsive to a connection request from a client machine to initiate a user session for associating a user session originating from a client machine with a given server in the pool in accordance with the load balancing routine;

means for returning a redirection response to the client machine for the connection request; and

means operative during the user session for redirecting to the given server any additional connection requests originating from the client machine.

19. The server as described in claim 18 wherein the means for associating comprises:

means for generating a virtual URL by modifying a given URL to include a session identifier;

means for redirecting a given connection request to the given server using the virtual URL.

20. The server as described in claim 19 wherein the redirector further includes:

means for inactivating the virtual URL upon completion of the user session.

21. A method of managing a pool of servers at a Web site identified by a given URL, comprising the steps of:

responsive to a connection request from a client machine to initiate a user session, associating a user session originating from a client machine with a server in the pool of servers in order to distribute user sessions across the pool of servers in accordance with a load balancing protocol; and

returning a redirection response to a given client machine for the connection request; and

during a given user session initiated from the given client machine, serving content to the given client machine only from its associated server.

22. A method of managing a pool of servers at a Web site identified by a given URL, comprising the steps of:

in response to a connection request containing the given URL from a given client machine that initiates a session, associating a session identifier with a given server in the pool of servers;

generating a virtual URL by modifying the given URL from the connection request to include the session identifier;

generating a redirection response comprising the virtual URL; and

sending the redirection response to the given client machine to redirect the connection request to the given server.